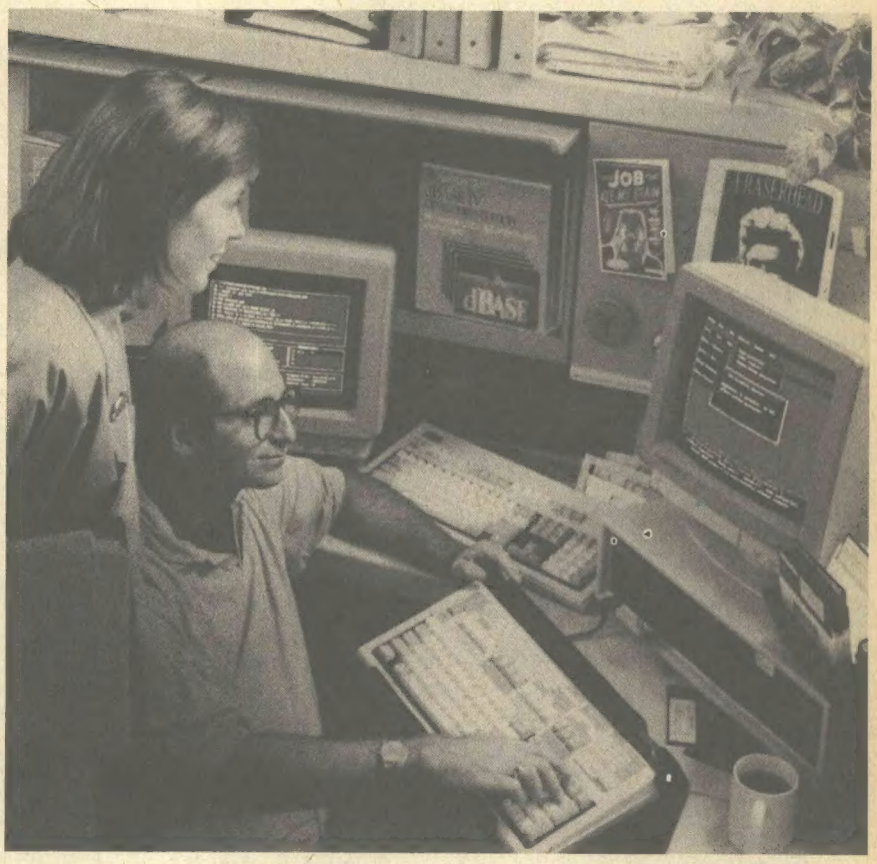
ขจร โรจนเมธินทร์

SOFT MAIL

Visual Basic 3.0 Professional

: สร้างโปรแกรมสกรีนเซฟเวอร์

แนะนำวิธีการเขียนแอพพลิเคชันประเภทสกรีนเซฟเวอร์แบบง่าย ๆ



หลังจากที่วิชวลเบสิก 3.0 ออกมาให้ใช้ได้ไม่นาน ตอนนี้เริ่มมีข่าวคราวของ VB 4.0 กันอีกแล้ว ซึ่งก็แน่นอนว่าคงจะมีสิ่งใหม่ ๆ อีกหลายอย่างมาให้เราทดลองใช้กัน คงต้องรอคอยกันอีกสักพัก ให้ซอฟต์แวร์เข้ามาในเมืองไทยก่อนจึงจะมีโอกาสนำมาฝากผู้อ่านกันบ้าง ตอนนี้ก็มาว่ากันถึงเรื่อง VB 3.0 ก่อน ในตอนนี้ผู้เขียนจะนำเอาวิธีการเขียนแอพพลิเคชันประเภทสกรีนเซฟเวอร์แบบง่าย ๆ มาให้ดูกัน ผู้อ่านที่ใช้งานวินโดวส์อยู่คงคุ้นเคยกับ Afterdark ซึ่งเป็นแอพพลิเคชันแบบนี้เป็นอย่างดีแล้ว หลายคนคงเคยสงสัยว่าโปรแกรมมีหลักการทำงานอย่างไร ผู้เขียนได้ทดลองเขียนขึ้นดูปรากฏว่าโปรแกรมทำงานได้ถูกต้องดี แต่ไม่ทราบว่าจะเป็นวิธีเดียวกับโปรแกรมที่ขายกันหรือไม่

สร้างฟอร์มหลักของโปรแกรม

จากรูปที่ 1 แสดงคอนโทรลที่เป็นส่วนประกอบของฟอร์มหลักของโปรแกรม ซึ่งประกอบด้วยคอนโทรลเลเบล 3 อัน ใช้ที่บนจอภาพ (Delay time) คอนโทรลสปินบัททอน ใช้กำหนดค่า Delay แสดงข้อความและแสดงค่าเวลาที่รอก่อน โปรแกรมจะแสดงภาพกราฟิกเคลื่อน time คอนโทรลไทม์เมอร์ซึ่งเป็นส่วนสำคัญของโปรแกรมที่คอยควบคุมการทำงานของโปรแกรมให้แสดงภาพ

รูปที่ 1

กราฟิกเมื่อถึงเวลาที่กำหนด คอนโทรลคอมมานด์บัททอนใช้สำหรับออกจากโปรแกรม และสุดท้ายเป็นคอนโทรล-ลิสต์บ็อกซ์ ใช้เก็บชื่อแบบการแสดงกราฟิกของโปรแกรม โดยในโปรแกรมนี้ผู้เขียนจะกำหนดไว้เพียงสามแบบเท่านั้น ผู้อ่านอาจจะนำไปเขียนเพิ่มเติมเองได้

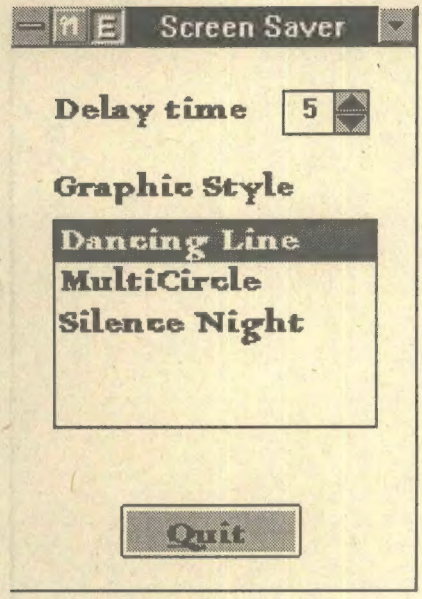
การกำหนดค่าเริ่มต้นให้กับพรอพ-เพอร์ตี้ส์ของฟอร์มและคอนโทรลที่จำเป็นนั้น ได้แสดงอยู่ในตารางที่ 1 แล้ว

ออปเจ็กต์	พรอพเพอร์ตี้ส์	ค่าเริ่มต้น
Form	Name	Form1
	AutoRedraw	True
	BackColor	&H00FFFF80& (Cyan)
	BorderStyle	1 'Fixed Single
	Caption	"Screen Saver"
	WindowState	1 'Minimized
ListBox	Name	lstGraph
	FontName	"Colonna MT"
	FontSize	13.5
	TabIndex	0
CommandButton	Name	cmdQuit
	Caption	"&Quit"
	FontName	"Colonna MT"
	FontSize	13.5
SpinButton	Name	Spin1
Timer	Name	Timer1
	Interval	1000
Label	Name	Label3
	BackStyle	0 'Transparent
	Caption	"Graphic Style"
	FontName	"Colonna MT"
	FontSize	13.5
	ForeColor	&H000000FF& (Red)
Label	Name	Label2
	Alignment	2 'Center
	BackStyle	0 'Transparent
	BorderStyle	1 'Fixed Single
	Caption	"5"
Label	Name	Label1
	BackStyle	0 'Transparent
	Caption	"Delay time"
	FontName	"Colonna MT"
	FontSize	13.5
	ForeColor	&H000000FF& (Red)

ตารางที่ 1 แสดงค่าเริ่มต้นของฟอร์มและคอนโทรลแต่ละชนิด

หลักการทำงานของโปรแกรม

การทำงานของโปรแกรมนี้จะอาศัยคอนโทรลไทม์เมอร์เป็นหลัก โดยไทม์เมอร์จะเป็นตัวนับเวลาว่าถึงช่วงเวลาที่ต้องแสดงภาพหรือยัง ซึ่งในระหว่างที่กำลังนับเวลาอยู่นั้นโปรแกรมจะต้องดักจับการทำงานของเม้าส์เอาไว้ก่อนด้วย เพราะว่าถ้าหากผู้ใช้มีการขยับเม้าส์ไปเวลาใดก็จะต้องกำหนดค่าของตัวแปรที่ใช้นับเวลาใหม่ และที่สำคัญต้องยกเลิกการดักจับเม้าส์ด้วย เพราะในระหว่างที่เราดักจับไว้นั้นเมสเสจทั้งหมดของเม้าส์จะถูกส่งมาที่โปรแกรมนี้ทั้งนั้น จึงต้องยกเลิกเพื่อให้วินโดวส์ส่งเมสเสจไปยังโปรแกรมที่ต้องการใช้ได้ถูกต้อง เมื่อถึงเวลาที่กำหนดเราก็จะต้องเก็บภาพของหน้าต่างที่แสดงอยู่บนจอภาพทั้งหมดไว้เพื่อจะนำกลับมาคืนภายหลังเมื่อมีการขยับเม้าส์ หลังจากเก็บรายละเอียดบนจอภาพไว้แล้วก็จะลบรายละเอียดทั้งหมดบนจอภาพออก แล้วเริ่มเขียนภาพกราฟิกที่ต้องการบนจอภาพไปจนกว่าจะมีการขยับเม้าส์ของผู้ใช้เมื่อใดก็จะคืนภาพของจอภาพเดิมและเริ่มนับเวลาใหม่อีกครั้ง



รูปที่ 2

```
Sub Form_Load ()
    blank = False
    counter = 0
    delayTime = 300
    gType = 0
    lstGraph.AddItem "Dancing Line"
    lstGraph.AddItem "MultiCircle"
    lstGraph.AddItem "Silence Night"
    lstGraph.ListIndex = 0
    xSize = GetSystemMetrics(SM_CXSCREEN)
    ySize = GetSystemMetrics(SM_CYSCREEN)
    SetRect rectScr, 0, 0, xSize, ySize
End Sub
```

รูปที่ 3
แสดงโปรแกรมส่วน Form_Load ที่ใช้กำหนดค่าเริ่มต้นต่าง ๆ

เตรียมการทำงานใน Form_Load

ขณะโปรแกรมเริ่มทำงาน (ดูรูปที่ 2) เราจะต้องเตรียมงานต่าง ๆ ตามโปรแกรมในรูปที่ 3 ได้แก่ การกำหนดค่าตัวแปรต่าง ๆ

blank ใช้เป็นแฟล็กสำหรับตรวจสอบว่าต้องเริ่มแสดงภาพกราฟิก

counter ใช้เป็นตัวนับเวลา

delayTime ใช้เก็บค่าช่วงเวลาก่อนที่จะเริ่มแสดงภาพกราฟิก โดยเริ่มที่ 300 วินาทีหรือ 5 นาที

gType ใช้เก็บแบบภาพกราฟิกที่เลือกให้แสดงบนจอภาพ เริ่มต้นเป็น 0 คือวาดเส้นตรงหลาย ๆ เส้นบนจอภาพ

จากนั้นจึงใส่เท็กซ์ให้กับลิสต์-บ็อกซ์ด้วยเมธอด AddItem 3 ข้อความ ได้แก่ Dancing Line, MultiCircle, Silence Night และกำหนดให้ค่า ListIndex เป็นศูนย์เพื่อให้แถบสีแสดงอยู่ที่รายการแรกของลิสต์บ็อกซ์ อ่านค่าความกว้างและความสูงของจอภาพมาเก็บที่ xSize, ySize ด้วยฟังก์ชัน GetSystemMetrics ของวินโดวส์ แล้วนำค่าที่ได้นี้มากำหนดค่าตัวแปร rectScr ซึ่งเป็นโครงสร้างชนิด RECT ใช้สำหรับเก็บขอบเขตของจอภาพที่ติดตั้งอยู่ โดยใช้ฟังก์ชัน SetRect

โพรซีเดอร์ Timer1_Timer และ Form_MouseMove

โพรซีเดอร์ทั้งสองนี้ถือว่าเป็นส่วนสำคัญที่สุดของโปรแกรม สำหรับ Timer1_Timer นั้นจะถูกเรียกมาทำงานทุก ๆ 1 วินาที ตามค่าพรอพเพอร์ตี้ส์ Interval ที่กำหนดไว้เป็น 1000 เมื่อเริ่มทำงานโปรแกรมจะดักจับการทำงานของเม้าส์ด้วย SetCapture จากนั้นก็ตรวจดูว่าขณะนั้นกำลังแสดงภาพกราฟิกอยู่หรือไม่ จากค่าแฟล็ก blank ซึ่งถ้าใช่ก็จะเรียกโพร-ซีเดอร์สำหรับวาดภาพมาทำงาน ในที่นี้ก็คือ drawLine, drawCircle, drawStar แล้วแต่กรณี ถ้าไม่ได้วาดภาพอยู่ก็จะตรวจว่าขณะนั้นถึงช่วงเวลาที่จะต้องเริ่มแสดงภาพกราฟิกหรือยัง ถ้าถึงแล้วก็จะกำหนดให้ค่า blank เป็นจริง เก็บรายละเอียดทั้งหมดบนจอภาพขณะนั้นด้วย saveOrgScr และลบภาพที่อยู่บนจอภาพด้วย blankScr กรณีที่ยังไม่ถึงเวลาก็จะนับค่าตัวแปร counter ต่อไปก่อน

โพรซีเดอร์ Form_MouseMove จะถูกเรียกมาทำงานเมื่อใดก็ตามที่ผู้ใช้ขยับเม้าส์ ทั้งนี้เนื่องจากเราดักจับการทำงานของเม้าส์อยู่ ดังนั้นตัวโปรแกรมจะได้รับเมสเสจทุกอันที่เกิดจากเม้าส์ เมื่อผู้ใช้ขยับเม้าส์โปรแกรมก็มีหน้าที่ที่จะกำหนดค่าตัวนับเวลาใหม่ทุกครั้งคือ กำหนดให้ counter เป็นศูนย์ กำหนดให้แฟล็ก blank เป็น false เพื่อให้ Timer1_Timer รู้ว่าขณะนั้นไม่ต้องแสดงภาพกราฟิก แล้วจากนั้นก็จะยกเลิกการดักจับเม้าส์เพื่อให้วินโดวส์ส่งเมสเสจของเม้าส์ไปให้โปรแกรมที่ต้องการได้ถูกต้องด้วย ReleaseCapture และสุดท้ายถ้าการขยับเม้าส์นั้นเป็นในกรณีที่เกิดหลังจากจอภาพถูกลบรายละเอียดไปด้วยเราก็ต้องคืนรายละเอียดเดิมของจอภาพด้วย restoreOrg-Scr ในกรณีนี้เราจะตรวจได้จากแฟล็ก clrScr ซึ่งจะถูกกำหนดให้เป็นจริงใน blankScr

```
Sub Timer1_Timer ()
    SetCapture hWnd
    If blank Then
       Select Case gType
          Case 0
            drawLine
          Case 1
            drawCircle
          Case 2
            drawStar
       End Select
    ElseIf counter = delayTime Then
       blank = True
       saveOrgScr
       blankScr
    Else
       counter = counter + 1
    End If
End Sub

Sub Form_MouseMove (Button As Integer, Shift As Integer, x As Single, y As Single)
    counter = 0
    blank = False
    ReleaseCapture
    If clrScr Then
       restoreOrgScr
    End If
End Sub
```

รูปที่ 4 แสดงโปรแกรมส่วนโพรซีเดอร์ Timer1_Timer และ Form_MouseMove

```
Sub blankScr ()
    Dim hDC As Integer
    Dim hBrush As Integer

    clrScr = True
    hDC = CreateDC("DISPLAY", 0, 0, 0)
    hBrush = CreateSolidBrush(RGB(0, 0, 0))
    SelectObject hDC, hBrush
    FillRect hDC, rectScr, hBrush
    DeleteDC hDC
    DeleteObject hBrush
End Sub
```

รูปที่ 5 แสดงโปรแกรมโพรซีเดอร์ส่วน blankScr

โพรซีเดอร์ saveOrgScr, blankScr, restore OrgScr

ผู้เขียนได้กล่าวถึงหน้าที่ของโพรซีเดอร์ทั้งสามไปบ้างแล้ว ต่อไปนี้ก็ลองดูขั้นตอนการทำงานของแต่ละโพรซีเดอร์ เริ่มที่ blankScr ซึ่งจะกำหนดค่า clrScr ให้เป็นจริงเพื่อแสดงว่าจอภาพถูกลบไป จากนั้นก็อ่านค่าดีไวซ์คอนเท็กซ์ของจอภาพด้วย CreateDC สร้างแปรงทาสีดำซึ่งใช้ฟังก์ชัน RGB ของวิชวลเบสิก กำหนดค่า R = 0, G = 0, B = 0 ด้วย CreateSolid Brush เมื่อได้ค่าแฮนเดิลของแปรงแล้วก็ต้องเรียกมาใช้งานในดีไวซ์คอนเท็กซ์ด้วย SelectObject ผู้เขียนใช้ฟังก์ชัน FillRect ในการระบายสีให้จอภาพทั้งหมดเป็นสีดำ ซึ่งจะทำให้รายละเอียดเดิมบนจอถูกลบไปหมด หลังจากนั้นก็ต้องยกเลิกการใช้แฮนเดิลแปรงทาสีและค่าดีไวซ์คอนเท็กซ์

โพรซีเดอร์ saveOrgScr มีหน้าที่เก็บรายละเอียดเดิมบนจอภาพก่อนที่จะถูกลบไว้ เริ่มแรกก็อ่านดีไวซ์คอนเท็กซ์ของจอภาพมาไว้ที่ hDC จากนั้นสร้างค่าที่เทียบเท่าขึ้นมาอีกหนึ่งตัวคือ hMemDC ด้วย CreateCompatibleDC สร้างออปเจ็กต์ชนิดบิตแม็พด้วยฟังก์ชัน Create CompatibleBitmap โดยเก็บค่าแฮนเดิลไว้ใน hBitmap และเลือกออปเจ็กต์นี้ให้กับดีไวซ์คอนเท็กซ์ hMemDC แล้วจึงก๊อบปี้รายละเอียดของจอภาพมาเก็บไว้ที่ hMemDC ด้วย StretchBlt จากนั้นจึงนำกราฟิกที่ได้ไปฝากไว้ในคลิปบอร์ดอีกครั้ง เพราะเราต้องยกเลิกการใช้ค่า hMemDC, hBitmap ก่อนออกจากโพรซีเดอร์ สำหรับขั้นตอนการย้ายกราฟิกไปไว้ในคลิปบอร์ดดูได้จากโปรแกรมในรูปที่ 6

โพรซีเดอร์ restoreOrgScr ทำงานตรงข้ามกับ saveOrgScr นั่น คือ กำหนดให้แฟล็ก clrScr เป็น false จากนั้นก็สร้างดีไวซ์คอนเท็กซ์ hDC, hMemDC เปิดคลิปบอร์ดเพื่ออ่านค่ากราฟิกที่เก็บไว้กลับมาเขียนคืนให้บนจอภาพด้วยฟังก์-

```
Sub saveOrgScr ()
    Dim hDC As Integer
    Dim hMemDC As Integer
    Dim hBitmap As Integer

    hDC = CreateDC("DISPLAY", 0, 0, 0)
    hMemDC = CreateCompatibleDC(hDC)
    hBitmap = CreateCompatibleBitmap(hDC, xSize, ySize)
    SelectObject hMemDC, hBitmap
    StretchBlt hMemDC, 0, 0, xSize, ySize, hDC, 0, 0, xSize, ySize, SRCCOPY
    OpenClipboard hWnd
    EmptyClipboard
    SetClipboardData CF_BITMAP, hBitmap
    CloseClipboard
    DeleteDC hMemDC
    ReleaseDC hBitmap, hDC
End Sub
```

รูปที่ 6 แสดงโปรแกรมโพรซีเดอร์ saveOrgScr

```
Sub restoreOrgScr ()
    Dim hDC As Integer
    Dim hBitmap As Integer
    Dim hMemDC As Integer

    clrScr = False
    hDC = CreateDC("DISPLAY", 0, 0, 0)
    hMemDC = CreateCompatibleDC(hDC)
    OpenClipboard hWnd
    hBitmap = GetClipboardData(CF_BITMAP)
    SelectObject hMemDC, hBitmap
    GetObjectGDI hBitmap, 14, bm
    SetStretchBltMode hDC, COLORONCOLOR
    StretchBlt hDC, 0, 0, xSize, ySize, hMemDC, 0, 0, bm.bmWidth,
bm.bmHeight, SRCCOPY
    CloseClipboard
    DeleteDC hMemDC
End Sub
```

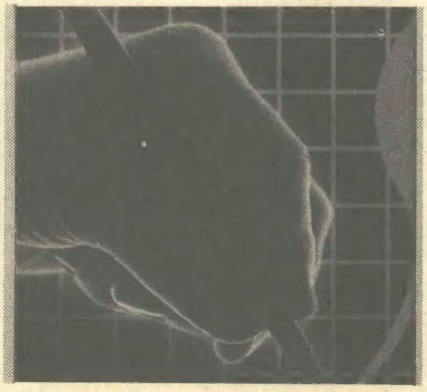
รูปที่ 7 แสดงโปรแกรมโพรซีเดอร์ restoreOrgScr

ชัน StretchBlt เช่นกัน แต่กรณีนี้ปลายทางเป็น hDC ของจอภาพแทน

โพรซีเดอร์ drawLine, drawCircle, drawStar

หน้าที่ของโพรซีเดอร์ทั้งสามนี้ก็คือ วาดภาพกราฟิกลงบนพื้นที่ว่างบนจอภาพ หลักการทำงานของโพรซีเดอร์จะต้องสุ่มหาตำแหน่งที่เหมาะสมซึ่งอยู่บนจอภาพมาใช้สำหรับวาดด้วยฟังก์ชัน Rnd ของวิชวลเบสิก โดยค่าของจุดต้องเป็นค่าเทียบกับไคล์เอ็นต์ไม่ใช่ค่าของสกรีน ดังนั้นจึงต้องใช้ฟังก์ชัน ScreenToClient มาแปลงก่อนเสมอ

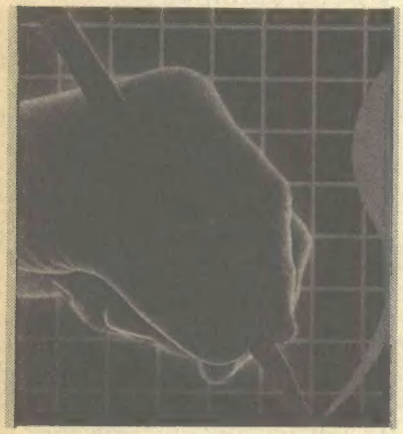
drawLine ในโพรซีเดอร์นี้เราจะสร้างปากกาสำหรับวาดภาพด้วย CreatePen โดยจะต้องสุ่มค่าสีของปากกาที่จะสร้างขึ้นมาก่อน จากนั้นจึงสุ่มค่าเพื่อใส่เป็นจุดเริ่มต้นของเส้นตรงแล้วใช้ MoveTo ย้ายไปอยู่ที่ตำแหน่งที่ได้ สุ่มค่าจุดปลายของเส้นตรงแล้วใช้ LineTo ลากเส้นตรงก่อนออกจากโพรซีเดอร์ต้องไม่ลืมลบแฮนเดิลของปากกาด้วย



drawCircle โพรซีเดอร์นี้จะวาดวงกลมบนจอภาพ โดยกำหนดจุดศูนย์กลางไว้ที่ค่า xPos, yPos และใช้ค่า yPos มากำหนดรัศมีของวงกลมต่อ กำหนดสีของแปรงที่ใช้ระบายในวงกลมเป็นสีดำด้วย CreateSolidBrush สร้างปากกาสำหรับวาดด้วยสีที่สุ่มมาด้วย CreatePen กำหนดชนิดของแบคกราวนด์เป็นแบบโปร่งใสด้วย SetBkMode จากนั้นจึงวาดวงกลมด้วยฟังก์ชัน Ellipse

drawStar โพรซีเดอร์นี้ทำงานง่าย ๆ โดยสุ่มสีที่จะใช้กำหนดให้พิกเซล

สุ่มตำแหน่งพิกเซลที่ต้องการแล้วใช้ SetPixel กำหนดค่าลงที่ตำแหน่งที่สุ่มได้



โพรซีเดอร์ cmdQuit_Click, lstGraph_click, Spin1_SpinDown, Spin1_SpinUp

cmdQuit_click มีหน้าที่เดียวคือ เมื่อผู้ใช้คลิ๊กเม้าส์ที่บัททอนก็เรียก End เพื่อจบโปรแกรม

lstGraph_Click มีหน้าที่นำค่าอินเด็กซ์ของลิสต์บ็อกซ์ lstGraph มากำหนดให้กับตัวแปร gType เมื่อผู้ใช้เลือกแบบของกราฟิกที่จะใช้วาดบนจอภาพจากลิสต์-บ็อกซ์

Spin1_SpinDown ใช้ลดค่าตัวแปร delayTime ลงครั้งละ 60 วินาที แล้วนำค่าไปแสดงที่ Caption ของเลเบล label2 ในหน่วยนาที โดยหารด้วย 60 สำหรับค่า delayTime นี้จะต้องอยู่ระ-หว่าง 1 ถึง 10 เท่านั้น

Spin1_SpinUp ทำงานตรงข้ามกับ SpinDown คือ เพิ่มค่าให้ delayTime ทีละ 60 วินาที และนำค่าไปแสดงที่ Caption ของเลเบล label2 ในหน่วยนาทีเช่นกัน

```
Sub dra wLine ()
    Dim hDC As Integer
    Dim hPen As Integer
    Dim R As Integer
    Dim G As Integer
    Dim B As Integer
    Dim xPos As Integer
    Dim yPos As Integer

    R = 255 * Rnd
    G = 255 * Rnd
    B = 255 * Rnd
    cpoint.x = xSize - 1
    cpoint.y = ySize - 1
    ScreenToClient hWnd, cpoint
    xPos = Int(cpoint.x * Rnd + 1)
    yPos = Int(cpoint.y * Rnd + 1)
    hDC = CreateDC("DISPLAY", 0, 0, 0)
    hPen = CreatePen(PS_SOLID, 1, RGB(R, G, B))
    SelectObject hDC, hPen
    MoveTo hDC, xPos, yPos
    xPos = Int(cpoint.x * Rnd + 1)
    yPos = Int(cpoint.y * Rnd + 1)
    LineTo hDC, xPos, yPos
    DeleteDC hDC
    DeleteObject hPen
End Sub

Sub drawCircle ()
    Dim hDC As Integer
    Dim hPen As Integer
    Dim hBrush As Integer
    Dim R As Integer
    Dim G As Integer
    Dim B As Integer
    Dim xPos As Integer
    Dim yPos As Integer
    Dim Radius As Integer

    R = 255 * Rnd
    G = 255 * Rnd
    B = 255 * Rnd
    cpoint.x = xSize - 1
    cpoint.y = ySize - 1
    ScreenToClient hWnd, cpoint
    xPos = cpoint.x / 2
```

รูปที่ 8 แสดงโปรแกรมส่วนที่ใช้วาดกราฟิกลงบนจอภาพ

รูปที่ 8 (ต่อ)

```
    yPos = cpoint.y / 2
    Radius = ((yPos / 2)) * Rnd
    hDC = CreateDC("DISPLAY", 0, 0, 0)
    hPen = CreatePen(PS_SOLID, 1, RGB(R, G, B))
    hBrush = CreateSolidBrush(RGB(0, 0, 0))
    SelectObject hDC, hPen
    SelectObject hDC, hBrush
    SetBkMode hDC, TRANSPARENT
    Ellipse hDC, xPos - Radius, yPos - Radius, xPos + Radius, yPos + Radius
    DeleteDC hDC
    DeleteObject hPen
    DeleteObject hBrush
End Sub

Sub drawStar ()
    Dim hDC As Integer
    Dim R As Integer
    Dim G As Integer
    Dim B As Integer
    Dim xPos As Integer
    Dim yPos As Integer

    R = 255 * Rnd
    G = 255 * Rnd
    B = 255 * Rnd
    cpoint.x = xSize - 1
    cpoint.y = ySize - 1
    ScreenToClient hWnd, cpoint
    xPos = cpoint.x * Rnd
    yPos = cpoint.y * Rnd
    hDC = CreateDC("DISPLAY", 0, 0, 0)
    SetPixel hDC, xPos, yPos, RGB(R, G, B)
    DeleteDC hDC
End Sub
```

```
Sub cmdQuit_Click ()
    End
End Sub

Sub lstGraph_Click ()
    gType = lstGraph.ListIndex
End Sub

Sub Spin1_SpinDown ()
    If delayTime > 60 Then
       delayTime = delayTime - 60
       label2.Caption = Str(delayTime \ 60)
    End If
End Sub

Sub Spin1_SpinUp ()
    If delayTime < 600 Then
       delayTime = delayTime + 60
       label2.Caption = Str(delayTime \ 60)
    End If
End Sub
```

รูปที่ 9
แสดงโพรซีเดอร์ส่วนอื่น ๆ ที่เหลือของโปรแกรม

```
Option Explicit

Global Const CF_BITMAP = 2
Global Const SRCCOPY = &HCC0020 ' (DWORD) dest = source
Global Const COLORONCOLOR = 3
Global Const SM_CXSCREEN = 0
Global Const SM_CYSCREEN = 0
Global Const TRANSPARENT = 1
Global Const PS_SOLID = 0
```

→

รูปที่ 10
แสดงโมดูลโปรแกรม BLANK.BAS ที่ใช้ประกาศตัวแปรและต้นแบบฟังก์ชันเอพีไอ

ก่อนจบ

ผู้เขียนต้องขอบอกก่อนว่าการเขียนโปรแกรมในบทความนี้เป็นแนวคิดคร่าว ๆ ในการสร้างโปรแกรมประเภทสกรีนเซฟเวอร์เท่านั้น ดังนั้นการทำงานบางจุดของโปรแกรมยังต้องการการปรับปรุงอีกมาก เช่น ผู้เขียนไม่ได้ซ่อนเม้าส์เคอร์เซอร์ขณะวาดกราฟิก ในช่วงดักจับเม้าส์นั้นการคลิ๊กเม้าส์จะเห็นว่าโปรแกรมตอบสนองช้าลงไปบ้าง แบบการวาดภาพมีน้อยเกินไป หากผู้อ่านท่านใดนำแนวคิดการเขียนโปรแกรมไปใช้ได้ผลประการใดก็กรุณานำมาฝากกันบ้าง

รูปที่ 10 (ต่อ)

```
Type POINTAPI
    X As Integer
    Y As Integer
End Type

Type RECT
    left As Integer
    top As Integer
    right As Integer
    bottom As Integer
End Type

Type BITMAP
    bmType As Integer
    bmWidth As Integer
    bmHeight As Integer
    bmWidthBytes As Integer
    bmPlanes As String * 1
    bmBitsPixel As String * 1
    bmBits As Long
End Type

Declare Sub SetCapture Lib "User" (ByVal hWnd As Integer)
Declare Sub ReleaseCapture Lib "User" ()

Declare Sub OpenClipboard Lib "User" (ByVal hWnd As Integer)
Declare Sub CloseClipboard Lib "User" ()
Declare Sub EmptyClipboard Lib "User" ()
Declare Sub SetClipboardData Lib "User" (ByVal wFormat As Integer, ByVal hMem As
Integer)
Declare Function GetClipboardData Lib "User" (ByVal wFormat As Integer) As Integer

Declare Function CreateDC Lib "GDI" (ByVal lpDriverName$, ByVal lpDeviceName&, ByVal
lpOutPut&, ByVal lpInitData&) As Integer
Declare Function CreateCompatibleDC Lib "GDI" (ByVal hDC As Integer) As Integer
Declare Sub DeleteDC Lib "GDI" (ByVal hDC As Integer)
Declare Function CreateCompatibleBitmap Lib "GDI" (ByVal hDC As Integer, ByVal nWidth
As Integer, ByVal nHeight As Integer) As Integer
Declare Sub ReleaseDC Lib "User" (ByVal hWnd As Integer, ByVal hDC As Integer)

Declare Sub SelectObject Lib "GDI" (ByVal hDC As Integer, ByVal hObject As Integer)
Declare Sub GetObjectGDI Lib "GDI" Alias "GetObject" (ByVal hObject As Integer, ByVal
nCount As Integer, lpObject As BITMAP)
Declare Sub DeleteObject Lib "GDI" (ByVal hObject As Integer)

Declare Sub StretchBlt Lib "GDI" (ByVal hDC%, ByVal X%, ByVal Y%, ByVal nWidth%, ByVal nHeight%,
ByVal hSrcDC%, ByVal XSrc%, ByVal YSrc%, ByVal nSrcWidth%, ByVal
nSrcHeight%, ByVal dwRop&)
Declare Sub SetStretchBltMode Lib "GDI" (ByVal hDC As Integer, ByVal nStretchMode As
Integer)

Declare Function CreateHatchBrush Lib "GDI" (ByVal nIndex As Integer, ByVal crColor As
Long) As Integer
Declare Function CreateSolidBrush Lib "GDI" (ByVal crColor As Long) As Integer
Declare Function CreatePen Lib "GDI" (ByVal nPenStyle As Integer, ByVal nWidth As
Integer, ByVal crColor As Long) As Integer

Declare Sub FillRect Lib "User" (ByVal hDC As Integer, lpRect As RECT, ByVal hBrush As
Integer)
Declare Sub SetPixel Lib "GDI" (ByVal hDC As Integer, ByVal X As Integer, ByVal Y As
Integer, ByVal crColor As Long)
Declare Sub Ellipse Lib "GDI" (ByVal hDC As Integer, ByVal X1 As Integer, ByVal Y1 As
Integer, ByVal X2 As Integer, ByVal Y2 As Integer)
Declare Sub MoveTo Lib "GDI" (ByVal hDC As Integer, ByVal X As Integer, ByVal Y As
Integer)
Declare Sub LineTo Lib "GDI" (ByVal hDC As Integer, ByVal X As Integer, ByVal Y As
Integer)

Declare Sub SetBkMode Lib "GDI" (ByVal hDC As Integer, ByVal nBkMode As Integer)
Declare Sub SetRect Lib "User" (lpRect As RECT, ByVal X1 As Integer, ByVal Y1 As Integer,
ByVal X2 As Integer, ByVal Y2 As Integer)
Declare Sub ScreenToClient Lib "User" (ByVal hWnd As Integer, lpPoint As POINTAPI)

Declare Function GetSystemMetrics Lib "User" (ByVal nIndex As Integer) As Integer
```